## REMARKS

Claims 1-41, 51-55 and 57-84 have been cancelled without prejudice. Claims 42-50 and 56 remain in the application for consideration. In view of the following remarks, Applicant respectfully requests that the rejections be withdrawn and for the application to be forwarded on to issuance.

### § 112 Rejections

Claims 42, 44, 46, 48, 50 and 56 stand rejected under 35 U.S.C. § 112, second paragraph as being indefinite. Specifically, the Office argues that "portions of memory" is vague, unclear, and does not have a well-defined meaning. In this regard, the Office states: "[f]or the purpose of examination, …'portion of memory' is interpreted by the Office as just a 'memory'".

Applicant respectfully disagrees and submits that "portions of memory" is not vague or unclear and that it has a well-defined meaning within the context of Applicant's disclosure. For example, the Office is directed to Figures 5 and 6 of Applicant's specification, which specifically illustrate an exemplary video (or graphics) card 500 and its corresponding on-board video memory 510 (including portions thereof), in accordance with one embodiment. The Office is also directed to page 16, line 9, through page 17, line 25, and page 19, line 4, through page 20, line 11, of Applicant's specification which, in pertinent part, describes the on-board video memory 510 (including portions thereof) as shown in Figures 5 and 6 respectively. These figures and excerpts are reproduced below for the Office's convenience (emphasis added):

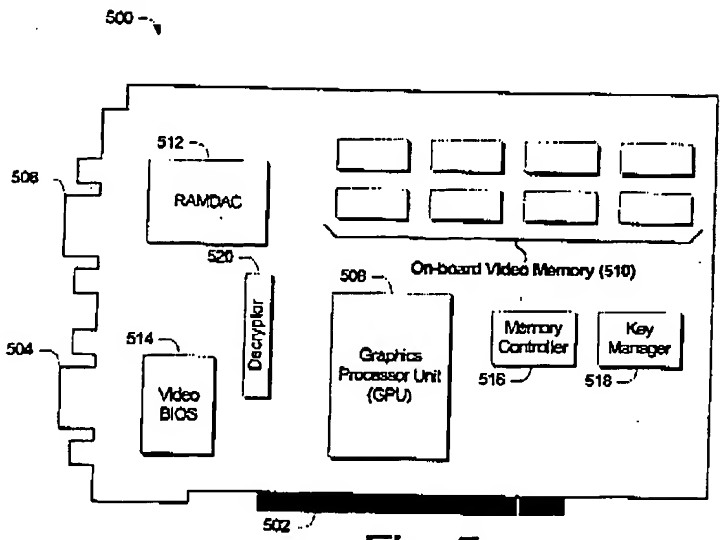

Fig. 5

**Exemplary First Video Card Embodiment**

***Fig. 5 shows an exemplary video (or graphics) card 500 in accordance with one embodiment.*** Card 500 includes a bus connector 502 that snaps into a port on a typical computer. Video card 500 also includes a monitor connector 504 (e.g. a 15-pin plug) that receives a cable that connects to a monitor. Video card 500 can, but need not, include a digital video-out (e.g. DVI) socket 506 that can be used for sending video images to digital displays and the like.

Like the video card of Fig. 1, video card 500 comprises a graphics processor unit (GPU) 508, ***video memory 510,*** random access memory digital-to-analogue converter (RAMDAC) 512, and driver software which can be included in the Video BIOS 514.

GPU 508 is a dedicated graphics processing chip that controls all aspects of resolution, color depth, and all elements associated with rendering images on the monitor screen. The memory controller (sometimes integrated into the GPU) manages the memory on the video card. The computer's central processing unit or CPU (not shown) sends a set of drawing instructions and data, which are interpreted by the graphics card's proprietary driver and executed by the card's GPU 508. GPU 508 performs such operations as bitmap transfers and painting, window resizing and repositioning, line drawing, font scaling and polygon drawing. The

GPU can then write the frame data to the frame buffer (or on-board video memory 510).

The information in the video memory frame buffer is an image of what appears on the screen, stored as a digital bitmap. RAMDAC 512 is utilized to convert the digital bitmap into a form that can be used for rendering on the monitor, as described above.

In addition to these components, in this embodiment, video card 500 comprises a memory controller 516 and a key manager 518. The video card can also include a decryptor 520. These components can be implemented in any suitable hardware, software, firmware or combination thereof.

Memory controller 516 receives encrypted data on the video card and decrypts the data into ***protected regions or portions of video memory 510.*** The decrypted data is now in a state in which it can be operated upon by the GPU 508. The memory controller can also be responsible for ensuring that data transfers on the video card are made between protected regions or regions that have a compatible degree of protection. That is, often times during processing of the data on the video card, the GPU 508 will operate on the data in the video memory (for example, by performing a blending operation) and will cause the resultant data to be written to a different video memory location. ***In this instance, there may be regions of video memory that are not protected or are protected at a different level.*** In this case, the memory controller can ensure that data transfers within the video card take place in a manner that ensures the protection of the unencrypted data. Examples of how this can be done are described below in more detail.

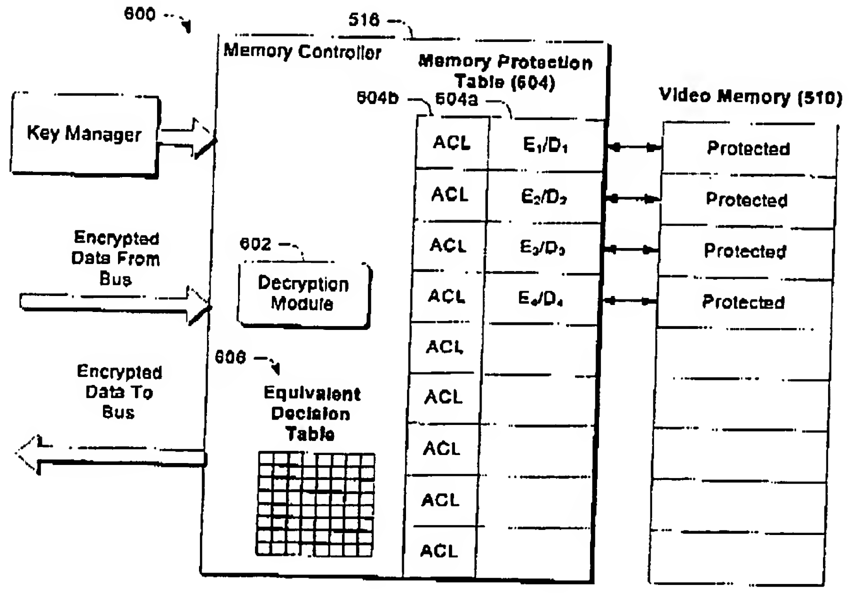

Fig. 6

***Fig. 6 shows, in accordance with one embodiment, selected components of video card 500 (Fig. 5)*** in more detail generally at 600. There, memory controller 516 comprises a decryption module 602 that is configured to receive encrypted data from the bus (either the PCI or AGP in this example) and decrypt the data into ***video memory 510.*** In addition, a memory protection table 604 and an equivalent decision table 606 are provided.

In this example, memory protection table 604 includes a table portion 604a that contains entries that associate encryption/decryption key pairs with ***individual portions of the video memory 510.*** For example, encryption/decryption key pair $E_1/D_1$ are associated with ***memory portion 608,*** encryption/decryption key pair $E_2/D_2$ are associated with memory portion 610 and so on. When decrypted data in the video memory is to be written to system memory off of the video card, or any other time when the data on the video card is to be provided over the bus (e.g. the PCI or AGP bus), the CPU can cause the data to be encrypted with one of the encryption keys in table portion 604a. The encrypted data can then be placed onto the bus and provided, for example, into the system memory. When the memory controller 516 receives encrypted data from the system memory that has been encrypted, for example, with key $E_1$, decryption module 602 can locate the associated decryption key $D_1$ and decrypt the data into memory portion 608.

*There can be a number of protected portions of video memory 510. In the present example, there are four protected portions designated 608-614. Unencrypted data in these protected portions can be processed by the GPU 508 (Fig. 5) in the usual manner. The protected portions of the video memory can be protected by an access control list. For example, memory protection table 604 includes a table portion 604b that can define an access control list for the various portions of video memory. Each portion of the video memory can have defined, in its associated access control list, which entities can access the memory portion. Thus, any attempted accesses by entities other than those contained in each memory portion's access control list will not be permitted by the memory controller.*

*Notice also that video memory 510 can include portions that are not protected. In this example, portions 616-624 are not protected. Accordingly, there are no encryption/decryption key pairs associated with these memory portions.*

As demonstrated by the above figures and excerpts, the subject matter with respect to "portions of memory" is clearly disclosed in Applicant's specification and has a well-defined meaning.

Furthermore, this particular claim language has been specifically endorsed by the Patent Office itself as evidenced from the number of patents (over 400) that have issued with claims drafted reciting the same or similar language. As an example, consider the following patents and the excerpted claims from each:

<u>6,909,384</u>

27. A method for preparing program data for delivery to a client that executes an electronic program guide, comprising:

initially allocating different-size *portions of memory* representative of a client memory for different time units represented in the electronic program guide;

evaluating whether program data for the different time units fits in the respective different-size *portions of memory*;

adjusting quantities of the program data for the different time units to identify an entire set of program data for storage at the client, wherein different quantities of the program data are stored for the different time units; and

compressing the entire set of program data by identifying frequently occurring character pairs in the program data and substituting character codes from a character code set, which are not used to represent individual characters, in place of the frequently occurring character pairs.

<u>**6,889,255**</u>

5. The method as recited in claim 1, wherein at least the performance monitoring data measurements are stored in one or more allocated ***portions of memory***.

17. The media of claim 7, wherein at least the performance monitoring data measurements are stored in one or more allocated ***portions of memory***.

<u>**6,785,790**</u>

1. A method for storing security attributes in a computer system, comprising: assigning security attributes to each of a plurality of ***portions of memory*** of a computer system; storing said security attributes in a multi-level lookup table having at least one first table and at least one second table, each of the first and second tables occupying one page; and storing at least a subset of said security attributes in a cache.

2. A method, as set forth in claim 1, wherein assigning security attributes to each of the plurality of ***portions of memory*** further comprises assigning security attributes to each page of memory of a computer system.

<u>**6,820,184**</u>

15. The digital system of claim 14 in which each of the plurality of algorithm modules further includes a memory interface, wherein the memory interface is operable to provide a set of memory usage requirements of the algorithm module to the framework and the method performed by the framework further comprises the step of allocating ***portions of memory*** to the algorithm module in accordance with the set of

memory usage requirements provided by the memory interface of the algorithm module.

**6,920,541**

1. A method for memory management in execution of a program by a computer having a memory, comprising:

allocating respective ***portions of the memory*** to data objects using mutator threads of the program, whereby the objects are held in a heap created by the program;

tracing the data objects in the heap so as to mark the data objects that are reachable at a given stage in the program;

looping over the mutator threads so as to verify for each of the mutator threads that every update to the allocated ***portions of the memory*** in progress by the mutator thread has been completed;

and sweeping the heap so as to free the memory that is allocated to the data objects that are not marked as reachable, for reallocation to new data objects.

In view of the language found in Applicant's specification and the number of issued patents with claims reciting "portions of memory", it is readily apparent that the language used in claims 42, 44, 46, 48, 50 and 56 is not unclear and has a well-defined meaning. Accordingly, Applicant traverses the Office's rejection.

**§102 Rejections**

Claims 42-50 and 56 stand rejected under 35 U.S.C. § 102(b) as being anticipated by U.S. Patent Application No. 6,064,739 to Davis (hereinafter "Davis").

**The Claims**

**Claim 42** recites a method comprising [emphasis added]:

- providing one or more key pairs, individual key pairs comprising an encryption key that can be used to encrypt data and a decryption key that can be used to decrypt data encrypted with the encryption key; and
- ***associating individual key pairs with individual portions of memory that comprise part of a video card memory.***

In making out the rejection of this claim, the Office argues that its subject matter is anticipated by Davis. Specifically, the Office first suggests that "frame data keys" shared between an encryptor and decryptor disclose "providing one or more key pairs", as claimed. The Office then relies on column 5, lines 42-44, and figure 3B (references "320", "324", and "302") of Davis as disclosing "associating individual key pairs with individual portions of memory that comprise part of a video card memory". In this regard, the Office states:

> "the frame date encryptor has encryption key which is used by the Frame Data Encryptor which is also associated with the memory/frame buffer shown on figure 3A or 3B reference '320' and figure 4, reference '424' and the corresponding decryption key which is used by frame Data Decryptor is also associated with the memory/frame buffer as shown on figure 3A/3B reference '324' and figure 4, reference '444'.

Applicant respectfully disagrees and submits that the Office has mischaracterized the Davis reference. Specifically, column 5, lines 42-44, of Davis merely discloses that the frame data encryptor shares frame data keys with the frame data decryptor. In this regard, a communication path between the encryptor and decryptor is utilized to transfer frame data keys, which are periodically changed to reduce the likelihood of a cryptographic attack. Column 5, lines 42-49, of Davis is reproduced below for the Office's convenience:

> In one embodiment, the frame data encryptor 320 shares "frame data keys" with a frame data decryptor 324, also located within the SVCP 302. Thus, a communication path 328 is needed between the frame data encryptor 320 and thc frame data decryptor 324 to transfer the frame data keys. It is contemplated that these "frame data keys" may be session keys which preferably are periodically changed to reduce the likelihood of a successful cryptographic analytic attack.

As is evident, from even a cursory inspection, the above excerpt does not disclose or suggest that frame data keys are provided *as pairs* or that individual key pairs are *associated* with individual *portions of memory*, as the Office suggests.

Nevertheless, in this regard, the Office reasons that since an encryption key in Davis is used by the encryptor (as shown in Fig. 3B) and the encryptor is associated with a frame buffer, that somehow the *key itself* is associated with the frame buffer. However, Davis simply does not disclose or suggest that the key itself is associated with individual portions of the frame buffer. Accordingly, the Office has no basis for concluding that the *key itself* is associated with the frame buffer.

Perhaps more importantly, the Davis reference fails to disclose or suggest *portions* of memory at all, as contemplated in this claim and described in Applicant's specification. Thus, even if the encryption or decryption keys in Davis were provided as pairs and were associated with the frame buffer, (which they are not), they could not possibly be associated with "individual portions of memory that comprise part of a video card memory", as claimed.

In this regard, to assist the Office in appreciating subject matter within the spirit of this claim, the Office is directed to figure 6 (reproduced above) which is one example of "portions of memory", as that term is understood in the context of Applicant's disclosure. As shown in fig. 6, video memory 510 can include protected portions (608-614) and unprotected portions (616-624). Furthermore, these individual portions comprise part of a video (or graphics) card's memory. In contrast, and as noted above, Davis fails to mention *portions* of memory or *portions* of a frame buffer at all.

Accordingly, since Davis does not disclose or suggest "providing one or more key pairs" or "associating individual key pairs with individual portions of memory that comprise part of a video card memory", this claim is allowable.

**Claims 43-50** depend from claim 42 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 42, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

In addition, Applicant is particularly puzzled by the Office's argument in regards to claims 44 and 45. Specifically, the Office relies on column 5, lines 42-49, and Fig. 3B (Frame Data Encryptor 320 and Frame Data Decryptor 324) of Davis in arguing that it discloses a "table/memory controller" on the video card having individual entries that associate individual key pairs, as claimed. However, even a cursory examination of this excerpt (reproduced above) indicates that "table" or "memory controller" is not even mentioned. In fact, these terms are not mentioned in the Davis reference at all. Accordingly, Applicant respectfully submits that the Office has mischaracterized Davis in this regard.

**Claim 56** recites a method comprising [emphasis added]:

- reading data from one ***or more portions of memory on a video card, individual portions of the memory having an associated encryption/decryption key pair***;
- recording ***key pairs associated with the memory portions*** from which the data was read;
- operating on the data read from the ***one or more portions of the memory*** to provide output data;
- ***ascertaining whether the key pairs associated with the memory portions from which the data was read are equivalent to a key pair associated with a video memory portion that is to serve as a destination for the output data;*** and
- if the key pairs are equivalent, providing the output data into the destination ***video memory portion***.

In making out the rejection of this claim, the Office argues that its subject matter is anticipated by Davis. Specifically, as discussed above, the Office improperly interprets "portion(s) of memory" or "memory portion(s)" as "just a 'memory'". Based on this interpretation, the Office relies on columns 5 and 6 of Davis as disclosing the subject matter recited in this claim. Regarding the first element, the Office essentially argues that data retrieved from the frame buffer by the IDD unit in Davis anticipates "reading data from one or more portions of memory", as claimed. Next, regarding the second element, the Office relies on the sharing of "frame data keys", as disclosed in columns 5 and 6 of Davis, in arguing that the keys are "inherently recorded to perform a bidirectional authentication". Regarding the fourth element, the Office appears to argue that Davis discloses "ascertaining whether the key pairs associated with the memory portions from which the data was read are equivalent" in so far as the encryption and decryption "frame data keys" in Davis must be the same in order for data retrieved from the buffer to be decrypted. The Office then relies on this reasoning in arguing that in Davis, decrypted data is subsequently outputted to the display/out; thus disclosing "providing the output data into the destination video memory portion", as recited in the fifth element of this claim.

Applicant respectfully disagrees and submits that the Office has mischaracterized the Davis reference. First, as discussed above, the Davis reference fails to disclose or suggest portions of memory at all, or individual portions of memory having an associated encryption/decryption key pair, as those terms and concepts are utilized in the claim and described in the Specification. As such, Davis could not possibly disclose or suggest *any* of the elements of this claim.

Furthermore, the Office's argument that the encryption and decryption "frame data keys" in Davis must be the same in order for data retrieved from the buffer to be decrypted is misplaced. Specifically, the Office's argument involves comparing individual single keys (an encryption key and corresponding decryption key) in order to determine whether they are "the same" so that encrypted data can be decrypted. However, the Office appears to ignore the fact that the fourth element of this claim recites: ascertaining whether *key pairs* (***not single keys***) associated with the *memory portions* (***Davis's keys have no such association***) from which the data was read are *equivalent* (***rather than just "the same"***) to a key pair associated with a video *memory portion* (***Davis's keys have no such association***) that is to serve as a destination for the output data ***(In Davis, the destination is the display/out)***.

In this regard, to further assist the Office in appreciating the distinction of "ascertaining whether the key pairs associated with the memory portions from which the data was read are equivalent", as claimed, the Office is directed to page 22, line 13, through page 23, line 8, of Applicant's specification - which provides one example of subject matter embodying the spirit of this claim. This excerpt is reproduced below for the Office's convenience [emphasis added]:

> In this example, protection consistency can be enforced by ***ascertaining that the keys associated with the memory portions that contained the input data are the same as or equivalent to the key associated with the memory portion that is to hold the output data.*** ***<u>Equivalence, in this example, can be determined based on the restrictions associated with the memory portions—that is—is there the same level of protection as between the different memory portions? Keys might not be equivalent if, for example, there are two different programs with two different pieces of content.</u>*** For example, say that each content is encrypted with a different key and that the programs associated with the keys cannot share content. In this example, the encryption keys are not equivalent.
>
> ***One way to ascertain the equivalence of keys associated with the various memory portions is to use an equivalent decision table such as table 606.*** There, the table defines a matrix that contains entries for each of the keys. An "X" in one of the matrix cells indicates that the keys are equivalent. For example, looking first at key $E_1$, this key is equivalent to keys $E_1$, $E_2$ and $E_4$. Thus, as shown best in Fig. 6, data can be freely transferred between memory portions 608, 610, and 614. Key $E_1$ is not, however, equivalent to key $E_3$. Thus, data cannot be transferred from memory portion 608 into memory portion 612. In the Fig. 8 example, ***since the input data comes from memory portions 608 and 610 and the resultant data is to be stored in memory portion 614, equivalency is not an issue and the operation can take place.***

The above discussion clearly shows that Davis does not disclose or suggest the subject matter of this claim. Accordingly, for at least this reason, this claim is allowable.

## Conclusion

All of the claims are in condition for allowance. Accordingly, Applicant requests a Notice of Allowability be issued forthwith. If the Office's next anticipated action is to be anything other than issuance of a Notice of Allowability, Applicant respectfully requests a telephone call for the purpose of scheduling an interview.

Respectfully Submitted,

Dated: 9/23/05

By: [illegible]
Lance R. Sadler
Reg. No. 38,605
(509) 324-9256